DOCKET NO. 98-MET-069C1 **PATENT**
CLIENT NO. STMI01-01012
Customer No. 30425

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	David L. Isaman
Serial No.:	09/443,160
Filed:	November 19, 1999
For:	SYMBOLIC STORE-LOAD BYPASS
Group No.:	2183
Examiner:	Daniel H. Pan

MAIL STOP APPEAL BRIEF - PATENTS
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

RESPONSE TO NOTICE OF NON-COMPLIANT APPEAL BRIEF

In response to the Notice of Non-Compliant Appeal Brief dated March 14, 2007, the Appellant respectfully requests a two (2) month extension of time. The period for replying to the Notice now expires on June 14, 2007.

In response to the Notice of Non-Compliant Appeal Brief, the Applicant is submitting a Substitute Appeal Brief. The Substitute Appeal Brief mainly provides additional material in the "Summary of Claimed Subject Matter" section.

The Appellant respectfully notes that 37 C.F.R. § 41.37(c)(1)(v) does not spell out the exact requirements for a "concise explanation of the subject matter defined in each of the independent claims." The Appellant respectfully submits that the Appellant's prior Appeal Brief

provided an adequate description of the invention as claimed in the independent claims. However, in order to move this application towards final decision on appeal, the Appellant is submitting the Substitute Appeal Brief, which includes additional information in the "Summary of Claimed Subject Matter" section. The Appellant respectfully submits that the Substitute Appeal Brief more than adequately presents a "concise explanation of the subject matter defined in each of the independent claims." The Appellant respectfully requests issuance of an Examiner's Answer responding the merits of the Appellant's Substitute Appeal Brief so that this application can proceed towards a final decision on appeal.

If any outstanding issues remain or if the Examiner has any suggestions for expediting allowance of this application, the Appellant respectfully invites the Examiner to contact the undersigned at the telephone number indicated below or at *dvenglarik@munckbutrus.com*.

The Commissioner is hereby authorized to charge any fees connected with this communication (including any extension of time fees) or credit any overpayment to Deposit Account No. 50-0208.

Respectfully submitted,

MUNCK BUTRUS, P.C.

Date: 6-14-2007

Daniel E. Venglarik
Registration No. 39,409

Docket Clerk
P.O. Box 802432
Dallas, Texas 75380
Tel: (972) 628-3600
Fax: (972) 628-3616
Email: *dvenglarik@munckbutrus.com*

DOCKET NO. 98-MET-069C1 **PATENT**
CLIENT NO. STMI01-01012
Customer No. 30425

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	David L. Isaman
Serial No.:	09/443,160
Filed:	November 19, 1999
For:	SYMBOLIC STORE-LOAD BYPASS
Technology Center:	2100
Group No.:	2183
Examiner:	Daniel H. Pan

MAIL STOP APPEAL BRIEF - PATENTS
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

SUBSTITUTE APPEAL BRIEF

In response to the Notice of Non-Compliant Appeal Brief dated March 14, 2007, the Applicant respectfully submits this Substitute Appeal Brief.

The Appellant has appealed to the Board of Patent Appeals and Interferences from the decision of the Examiner dated January 19, 2006, finally rejecting Claims 2-7, 12-17, 20, and 21. The Appellant filed a Notice of Appeal on April 19, 2006, which was received by the U.S. Patent and Trademark Office on April 25, 2006. The Appellant respectfully submits this brief on appeal.

TABLE OF CONTENTS

TABLE OF AUTHORITIES

Real Party in Interest

This application is currently owned by STMicroelectronics, Inc. as indicated by:

(1) an assignment recorded on January 24, 2000 in the Assignment Records of the United States Patent and Trademark Office at Reel 010517, Frame 0988; and

(2) a merger recorded on August 2, 2001 in the Assignment Records of the United States Patent and Trademark Office at Reel 012036, Frame 0306.

Related Appeals or Interferences

None - there are no appeals or interferences that will directly affect, be directly affected by, or have a bearing on the Board's decision in this pending appeal.

Status of Claims

Claim 1 has been cancelled. Claims 2-7, 12-17, 20, and 21 have been rejected pursuant to a final Office Action dated January 19, 2006. Claims 8-11, 18, and 19 have been objected to as being allowable but depending from rejected base claims pursuant to the final Office Action dated January 19, 2006. Claims 2-7, 12-17, 20, and 21 are presented for appeal. A copy of all pending claims is provided in Appendix A.

Status of Amendments after Final

The Appellant filed an AMENDMENT AND RESPONSE TO OFFICE ACTION on March 20, 2006. The Examiner refused to enter the AMENDMENT AND RESPONSE, asserting that it raised new issues that would require further consideration and/or search.

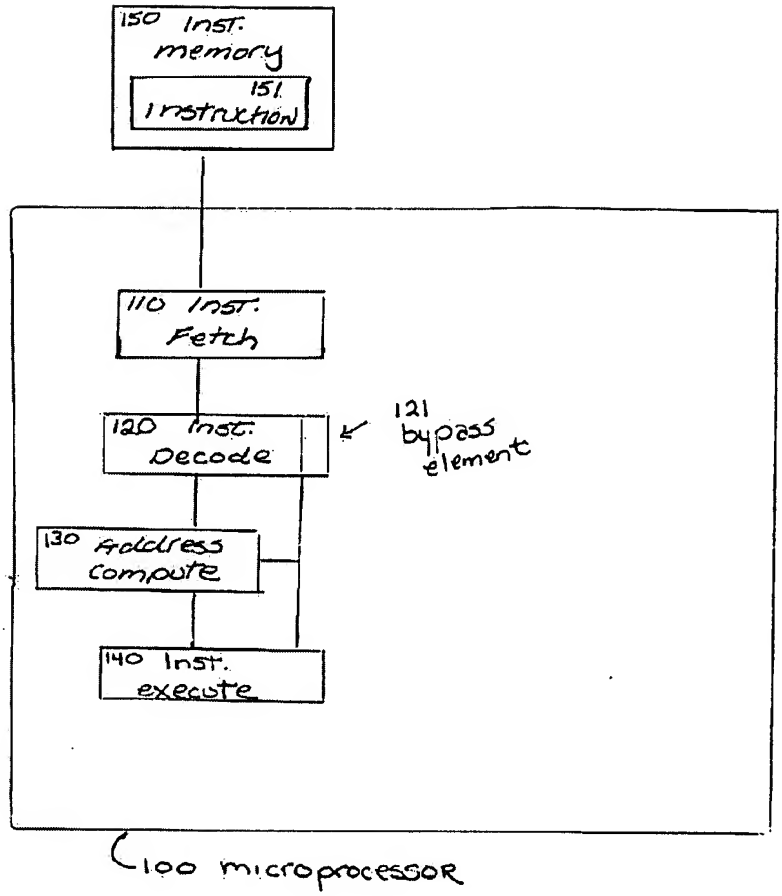
SUMMARY OF CLAIMED SUBJECT MATTER

The following summary refers to disclosed embodiments and their advantages but does not delimit any of the claimed inventions.

In General

The present application is directed, in general, to a method and system for operating a pipelined microprocessor more quickly. This can be accomplished by detecting instructions that load data from identical memory locations where data was recently stored. This detection may involve examining the symbolic structure of instructions as they are encountered and identifying identical memory locations based on their symbolic structure. The instructions can be detected without actually computing the external memory addresses referenced by the instructions. (*Application, Page 4, Lines 8-16*).

One example embodiment of the pipelined microprocessor is shown in Figure 1, which is reproduced below. In this examplary embodiment, a microprocessor 100 includes an instruction fetch stage 110, an instruction decode stage 120, an address computation stage 130, and an instruction execution stage 140. The instruction fetch stage 110 reads a sequence of instructions 151 from a memory 150. The instruction decode stage 120 parses the instructions 151, which may include determining the syntax of any external memory addresses of operands in the instructions 151. The address computation stage 130 can compute effective reference addresses for the instructions 151. (*Application, Page 6, Line 6 – Page 8, Line 18*).



The instruction decode stage 120 includes a bypass element 121. The bypass element 121 may generate a bypass signal when two instructions 151 refer to the same external memory location. The bypass element 121 may detect this condition by comparing base address and offset address values for two instructions 151. When the bypass signal is generated, the address computation stage 130 does not have to compute the actual external memory address for the microprocessor 100 to act on the knowledge that the instructions 151 refer to identical external memory addresses. (*Application, Page 8, Line 20 – Page 9, Line 11*).

One example use of this functionality occurs when a store instruction (storing data to an

external memory address) is followed by a load instruction (loading data from the same external memory address). In this example, the bypass signal may allow the load instruction to be performed using data transferred from an internal register, without requiring the microprocessor 100 to physically perform the load instruction. The store instruction is still physically performed by computing the effective reference address for the store instruction and physically storing data in the external memory. However, the subsequent load instruction can be completed without reference to the external memory. If the bypass signal had not been generated, the load instruction would be performed and completed by the external memory. (*Application, Page 12, Line 17 – Page 14, Line 8*).

Support for Independent Claims

Note that, per 37 C.F.R. § 41.37, only the independent claims are discussed in this section. The discussion of the claims in this section is for illustrative purposes and is not intended to affect the scope of the claims.

Claim 2

Regarding Claim 2, a pipelined microprocessor 100 is capable of detecting an instruction 151 that loads data from a first memory location that was previously stored to. (*Application, Page 8, Line 20 – Page 9, Line 7; Page 9, Lines 13-22; Page 11, Line 8 – Page 12, Line 11*). The instruction 151 is detected without requiring computation of an external memory address of the first memory location for the instruction 151. (*Application, Page 4, Lines 8-16; Page 8, Line 20 – Page 9, Line 7; Page 9, Lines 13-22; Page 11, Line 8 – Page 12, Line 11*).

As noted above, in one exemplary embodiment, the instruction decode stage 120 in a pipelined microprocessor 100 includes a bypass element 121, which may generate a bypass signal when two instructions 151 refer to the same external memory location. When the bypass signal is generated, the address computation stage 130 in the pipelined microprocessor 100 does not have to compute the actual external memory address for the microprocessor 100 to act on the knowledge that the instructions 151 refer to identical external memory addresses. (*Application, Page 8, Line 20 – Page 9, Line 11*).

One example use of this functionality occurs when a store instruction (storing data to an external memory address) is followed by a load instruction (loading data from the same external memory address). In this example, the bypass signal may allow the load instruction to be performed using data transferred from an internal register, without requiring the microprocessor 100 to physically perform the load instruction. (*Application, Page 12, Line 17 – Page 14, Line 8*).

Therefore, as recited in Claim 2, a pipelined microprocessor 100 detects an instruction 151 that loads data from a first memory location that was previously stored to, such as when the bypass element 121 detects a load instruction that loads data from the same external memory address used by a previous store instruction. Moreover, as recited in Claim 2, the pipelined microprocessor 100 can detect the instruction 151 without requiring computation of an external memory address of the first memory location for the instruction 151, such as when the bypass element 121 detects the load instruction without using any effective reference addresses output from the address computation stage 130.

Claim 12

Regarding Claim 12, a method for operating a pipelined microprocessor 100 includes detecting in the pipelined microprocessor 100 an instruction 151 that loads data from a first memory location that was previously stored to. (*Application, Page 4, Lines 8-16; Page 8, Line 20 – Page 9, Line 7; Page 9, Lines 13-22; Page 11, Line 8 – Page 12, Line 11*). The instruction 151 is detected without requiring computation of an external memory address of the first memory location for the instruction 151. (*Application, Page 4, Lines 8-16; Page 8, Line 20 – Page 9, Line 7; Page 9, Lines 13-22; Page 11, Line 8 – Page 12, Line 11*).

Once again, in an exemplary embodiment, the instruction decode stage 120 in a pipelined microprocessor 100 includes a bypass element 121, which may generate a bypass signal when two instructions 151 refer to the same external memory location. When the bypass signal is generated, the address computation stage 130 in the pipelined microprocessor 100 does not have to compute the actual external memory address for the microprocessor 100 to act on the knowledge that the instructions 151 refer to identical external memory addresses. (*Application, Page 8, Line 20 – Page 9, Line 11*).

One example use of this functionality occurs when a store instruction (storing data to an external memory address) is followed by a load instruction (loading data from the same external memory address). In this example, the bypass signal may allow the load instruction to be performed using data transferred from an internal register, without requiring the microprocessor 100 to physically perform the load instruction. (*Application, Page 12, Line 17 – Page 14, Line 8*).

Therefore, as recited in Claim 12, a method for operating a pipelined microprocessor 100 includes detecting an instruction 151 that loads data from a first memory location that was previously stored to, such as by using the bypass element 121 to detect a load instruction that loads data from the same external memory address used by a previous store instruction. Moreover, as recited in Claim 12, the pipelined microprocessor 100 is operated to detect the instruction 151 without requiring computation of an external memory address of the first memory location for the instruction 151, such as when the bypass element 121 detects the load instruction without using any effective reference addresses output from the address computation stage 130.

Claim 20

Regarding Claim 20, a method for operating a pipelined microprocessor 100 includes detecting a first instruction 151 that stores data to a first memory location, where the first instruction includes syntax for computing an effective address for the first memory location. (*Application, Page 7, Lines 8-21; Page 9, Lines 1-18*). The method also includes detecting a second instruction that loads data from a second memory location, where the second instruction includes syntax for computing an effective address for the second memory location. (*Application, Page 7, Lines 8-21; Page 9, Lines 1-18*). The method further includes determining the syntax for the first instruction and the syntax for the second instruction. (*Application, Page 7, Lines 8-21; Page 9, Lines 1-18; Page 12, Lines 8-15*). The method also includes using the syntax for the first instruction and the syntax for the second instruction to determine a

relationship between the first memory location and the second memory location, without requiring computation of the effective address for the first memory location and without requiring computation of the effective address for the second memory location. (*Application, Page 4, Lines 8-16; Page 8, Line 20 – Page 9, Line 7; Page 9, Lines 13-22; Page 11, Line 8 – Page 12, Line 11*). In addition, the method includes using the relationship to determine whether to perform one of the first instruction and the second instruction. (*Application, Page 10, Lines 1-5; Page 14, Lines 1-4*).

Once again, in an exemplary embodiment, the instruction decode stage 120 in a pipelined microprocessor 100 includes a bypass element 121, which may generate a bypass signal when two instructions 151 refer to the same external memory location. The bypass element 121 may detect this condition by comparing base address and offset address values for two instructions 151. When the bypass signal is generated, the address computation stage 130 in the pipelined microprocessor 100 does not have to compute the actual external memory address for the microprocessor 100 to act on the knowledge that the instructions 151 refer to identical external memory addresses. (*Application, Page 8, Line 20 – Page 9, Line 11*).

One example use of this functionality occurs when a store instruction (storing data to an external memory address) is followed by a load instruction (loading data from the same external memory address). In this example, the bypass signal may allow the load instruction to be performed using data transferred from an internal register, without requiring the microprocessor 100 to physically perform the load instruction. (*Application, Page 12, Line 17 – Page 14, Line 8*).

Therefore, as recited in Claim 20, a method for operating a pipelined microprocessor 100 includes detecting a first instruction 151 that stores data to a first memory location and detecting a second instruction 151 that loads data from a second memory location. Also, the pipelined microprocessor 100 is operated to determine the syntaxes for the instructions and to determine a relationship between the first and second memory locations without requiring computation of the effective address for the first and second memory locations. This can be done, for example, by the bypass element 121 identifying the base address and offset address values for two instructions 151 and using these values to determine if the instructions store data to and load data from the same memory location. Moreover, as recited in Claim 20, this can be done without requiring computation of the effective addresses for the first and second memory locations, such as when the bypass element 121 detects the instructions without using any effective reference addresses output from the address computation stage 130. In addition, as recited in Claim 20, once the instructions are detected, the pipelined microprocessor 100 can determine whether to perform one of the instructions, such as by not physically performing the load instruction to load data from the memory location.

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

1. Are Claims 2-7 and 12-17 anticipated by U.S. Patent No. 5,475,823 to Amerson et al. ("*Amerson*")?

2. Are Claims 2 and 12 obvious over U.S. Patent No. 5,854,921 to Pickett ("*Pickett*") in view of *Amerson*?

3. Are Claims 2-7 and 12-17 obvious over *Amerson* in view of *Pickett*?

4. Are Claims 2-7 and 12-17 obvious over *Amerson* in view of U.S. Patent No. 5,706,224 to Srinivasan et al. ("*Srinivasan*")?

5. Are Claims 20 and 21 obvious over U.S. Patent No. 5,850,138 to Engebretsen et al. ("*Engebretsen*") in view of U.S. Patent No. 5,615,357 to Ball ("*Ball*")?

ARGUMENT

Stated Grounds of Rejection

Claims 2-7 and 12-17 stand rejected under 35 U.S.C. § 102(b) as being anticipated by U.S. Patent No. 5,475,823 to Amerson et al. ("*Amerson*"). Claims 2 and 12 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 5,854,921 to Pickett ("*Pickett*") in view of *Amerson*. Claims 2-7 and 12-17 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over *Amerson* in view of *Pickett*. Claims 2-7 and 12-17 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over *Amerson* in view of U.S. Patent No. 5,706,224 to Srinivasan et al. ("*Srinivasan*"). Claims 20 and 21 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 5,850,138 to Engebretsen et al. ("*Engebretsen*") in view of U.S. Patent No. 5,615,357 to Ball ("*Ball*").

Legal Standards

A prior art reference anticipates a claimed invention under 35 U.S.C. § 102 only if every element of the claimed invention is identically shown in that single reference, arranged as they are in the claims. (*MPEP § 2131; In re Bond*, 910 F.2d 831, 832, 15 U.S.P.Q.2d 1566, 1567 (Fed. Cir. 1990)). Anticipation is only shown where each and every limitation of the claimed invention is found in a single prior art reference. (*MPEP § 2131; In re Donohue*, 766 F.2d 531, 534, 226 U.S.P.Q. 619, 621 (Fed. Cir. 1985)).

Under 35 U.S.C. §103, the "scope and content of the prior art are to be determined;

differences between the prior art and the claims at issue are to be ascertained; and the level of ordinary skill in the pertinent art resolved. Against this background, the obviousness or nonobviousness of the subject matter is determined." (*Graham v. John Deere Co.*, 383 U.S. 1, 148 U.S.P.Q. 459, 467 (1966)). Obviousness cannot be inferred from a combination of references without a showing that one of ordinary skill would have been motivated to combine those references. When "prior art references require selective combination ... to render obvious a subsequent invention, there must be some reason for the combination other than the hindsight gained from the invention itself.... Something in the prior art as a whole must suggest the desirability, and thus the obviousness, of making the combination." (*Uniroyal, Inc. v. Rudkin-Wiley Corp.*, 5 U.S.P.Q.2d 1434, 1438 (Fed. Cir. 1988), *quoting Interconnect Planning Corp. v. Feil*, 227 U.S.P.Q. 543 (Fed. Cir. 1985) and *Lindemann Maschinenfabrik GmbH v. American Hoist & Derrick*, 221 U.S.P.Q. 481 (Fed. Cir. 1984)).

Analysis of Examiner's Rejections

The cited references are briefly discussed in relevant part for the appropriate rejections, and each rejection is addressed separately below.

Ground of Rejection 1: Claims 2-7 and 12-17 stand rejected under 35 U.S.C. § 102(b) as being anticipated by U.S. Patent No. 5,475,823 to Amerson et al. ("*Amerson*")

Claims 2-7 and 12-17

Claim 2 recites:

> [a] pipelined microprocessor capable of detecting an instruction that loads data from a first memory location that was previously stored to, wherein the instruction is detected without requiring computation of an external memory address of said first memory location for the instruction.

The Examiner fails to establish that *Amerson* anticipates all elements of Claim 2. In particular, the Examiner fails to establish that *Amerson* anticipates a microprocessor capable of detecting an instruction "without requiring computation of an external memory address of [a] first memory location for the instruction."

Amerson recites a memory processor that prevents errors when a compiler advances load instructions in a sequence of instructions. (*Abstract*). The processor intercepts all load and store instructions before the instructions enter a memory pipeline. (*Abstract*). The processor stores a load instruction for a particular period of time, which allows the processor to determine if a store instruction to the same address would have been executed before the load instruction. (*Abstract*). If a store instruction would have been executed, the processor uses the data from the store instruction for the load instruction. (*Abstract*). As part of the processor's operation, an address comparator 28 compares the memory address specified in a store instruction with memory addresses specified in load instructions. (*Col. 5, Lines 30-33*). In other embodiments, an address comparator 528 compares memory addresses from all store instructions to the memory addresses

from load instructions to "check for partial or complete overlap of the memory locations accessed by the load and store instructions." (*Col. 8, Lines 28-33*).

In every embodiment of *Amerson*, the processor of *Amerson* compares the actual memory addresses being accessed by load and store instructions. More specifically, each address comparator in *Amerson* compares the actual memory addresses being accessed by load and store instructions. The Examiner asserts that *Amerson* does not explicitly disclose calculating the actual memory addresses of load and store instructions. (*01/19/06 Office Action, Page 5, Section 11*). However, *Amerson* clearly discloses that the "memory addresses" are the locations in memory where data is read from or written to. (*See, e.g., Col. 1, Line 31 – Col. 2, Line 39*). The Examiner provides no explanation or evidence from *Amerson* showing that these "memory addresses" are anything other than actual memory addresses in a memory.

It is clear that *Amerson* operates by comparing external memory addresses. Because of this, it is also clear that *Amerson* requires computation of external memory addresses. The external memory addresses for load and store instructions must be computed before the address comparator of *Amerson* can compare the memory addresses. As a result, *Amerson* fails to anticipate detecting an instruction that loads data from a "first memory location ... without requiring computation of an external memory address of [the] first memory location for the instruction" as recited in Claim 2.

In order to show that *Amerson* still anticipates Claim 2 despite this deficiency, the Examiner attempts to distinguish between address computation and instruction detection in *Amerson*. The Examiner asserts that detecting instructions in *Amerson* involves comparing

addresses, not computing addresses. As a result, the Examiner asserts that while the memory addresses in *Amerson* may be computed, the memory addresses are not computed during the comparison of those addresses. (*See, e.g., 01/19/06 Office Action, Page 4, Section 10*). In other words, the Examiner asserts that actual memory addresses can be computed and compared in *Amerson* and still anticipate Claim 2 because the actual memory addresses are computed before they are compared.

This position is completely illogical. It basically allows the Examiner to argue that a reference requiring computation of external memory addresses can anticipate a claim specifically reciting that computation of external memory addresses is not required. Claim 2 is crystal clear – an instruction that loads data from a first memory location is detected "without requiring computation of an external memory address of [the] first memory location for the instruction." In order to detect memory instructions in *Amerson*, *Amerson* first must compute the actual memory addresses. *Amerson* cannot possibly compare two actual memory addresses without first computing the actual memory addresses. As a result, *Amerson* clearly requires computation of the actual memory addresses in order to detect the memory instructions.

The Examiner asserts that whether "the address had been computed or not is irrelevant to the claimed invention (applicant did not claim that the address must not have been computed...)." (*01/19/06 Office Action, Page 4, Section 10*). This position contradicts the express language of Claim 2, which specifically requires that an instruction associated with a first memory location be detected "without requiring computation of an external memory address of [the] first memory location." In order to compare the actual memory addresses in *Amerson*,

the actual memory addresses must be computed. The Examiner cannot possibly show that *Amerson*, a reference that requires computation of an actual memory address, can anticipate a claim that specifically says computation of the memory address is not required.

For these reasons, *Amerson* fails to anticipate the Appellant's invention as recited in Claim 2 (and its dependent claims). For similar reasons, *Amerson* fails to anticipate the Appellant's invention as recited in Claim 12 (and its dependent claims).

Accordingly, the Appellant respectfully requests that the § 102 rejection of Claims 2-7 and 12-17 be withdrawn and that Claims 2-7 and 12-17 be passed to allowance.

Ground of Rejection 2: Claims 2 and 12 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 5,854,921 to Pickett ("*Pickett*") in view of *Amerson*

Claims 2-7 and 12-17

Claim 2 recites:

> [a] pipelined microprocessor capable of detecting an instruction that loads data from a first memory location that was previously stored to, wherein the instruction is detected without requiring computation of an external memory address of said first memory location for the instruction.

The Examiner fails to establish that the proposed *Pickett-Amerson* combination discloses, teaches, or suggests all elements of Claim 2. In particular, the Examiner fails to establish that the proposed *Pickett-Amerson* combination discloses, teaches, or suggests a microprocessor capable of detecting an instruction "without requiring computation of an external memory address of [a] first memory location for the instruction."

Pickett recites a data prediction structure for use in microprocessors. (*Abstract*). The structure stores base addresses and "stride values," which are added to form prediction addresses. (*Col. 2, Lines 37-42*). The prediction addresses are then used to fetch data from a memory. (*Col. 2, Lines 42-45*). Instructions referencing operands in registers can retrieve the operands before entering a processing pipeline since no address calculation is needed to locate the operands. (*Col. 2, Lines 26-30*).

The Examiner acknowledges that *Pickett* fails to disclose the "detection of instructions as [claimed]." (*08/08/05 Office Action, Page 2, Section 3*). As shown above, *Amerson* requires computation of external memory addresses for load and store instructions in order to detect "an

instruction that loads data from a first memory location that was previously stored to." As a result, *Amerson* also fails to disclose the detection of instructions as claimed in Claim 2.

For these reasons, the proposed *Pickett-Amerson* combination fails to disclose, teach, or suggest the Appellant's invention as recited in Claim 2. For similar reasons, the proposed *Pickett-Amerson* combination fails to disclose, teach, or suggest the Appellant's invention as recited in Claim 12.

Accordingly, the Appellant respectfully requests that the § 103 rejection of Claims 2 and 12 be withdrawn and that Claims 2 and 12 be passed to allowance.

Ground of Rejection 3: Claims 2-7 and 12-17 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over *Amerson* in view of *Pickett*

Claims 2-7 and 12-17

Claim 2 recites:

> [a] pipelined microprocessor capable of detecting an instruction that loads data from a first memory location that was previously stored to, wherein the instruction is detected without requiring computation of an external memory address of said first memory location for the instruction.

The Examiner fails to establish that the proposed *Amerson-Pickett* combination discloses, teaches, or suggests all elements of Claim 2. In particular, the Examiner fails to establish that the proposed *Amerson-Pickett* combination discloses, teaches, or suggests a microprocessor capable of detecting an instruction "without requiring computation of an external memory address of [a] first memory location for the instruction."

In every embodiment of *Amerson*, the processor compares the actual memory addresses being accessed by load and store instructions. More specifically, the address comparators in *Amerson* compare the memory addresses being accessed by load and store instructions. As a result, *Amerson* fails to disclose the detection of instructions as claimed in Claim 2.

Regarding *Pickett*, the Examiner makes inconsistent arguments regarding the teachings of *Pickett*. As noted above, the Examiner acknowledges that *Pickett* fails to show the "detection of instructions as [claimed]" in Claim 2. (*08/08/05 Office Action, Page 2, Section 3*). The Examiner inconsistently asserts later that *Pickett* discloses a system that could detect instructions "without

the need of calculating the memory address." (*08/08/05 Office Action, Pages 6-7, Section 16*). These positions regarding *Pickett* are completely inconsistent – both cannot possibly be true.

Not only that, the portions of *Pickett* cited by the Examiner (column 2, lines 2-34 and column 8, lines 64-65) contain absolutely no mention of detecting an "instruction that loads data from a first memory location that was previously stored to ... without requiring computation of an external memory address of said first memory location for the instruction" as recited in Claim 2.

The first portion (column 2, lines 2-34) of *Pickett* simply recites how instructions with operands stored in memory may require multiple clock cycles to be executed, while instructions with operands stored in registers may require a single clock cycle to be executed. This portion of *Pickett* does recite that address calculation is not required for instructions to retrieve data from registers. However, this portion of *Pickett* in no way recites detecting an instruction that loads data from a first memory location that was previously stored to, where the instruction is detected "without requiring computation of an external memory address of said first memory location for the instruction" as recited in Claim 2.

Similarly, the second portion (column 8, lines 64-65) of *Pickett* simply recites that an operand value is provided to a particular unit via a load/store unit 222 if the operand value is retrieved from a memory location. This portion of *Pickett* says absolutely nothing about detecting an instruction "without requiring computation of an external memory address of said first memory location for the instruction" as recited in Claim 2.

In effect, the Examiner has simply shown that *Pickett* mentions retrieving data from memory, while another portion of *Pickett* mentions not calculating memory addresses for values stored in registers. However, none of the cited portions of *Pickett* disclose, teach, or suggest "detecting an instruction that loads data from a first memory location that was previously stored to ... without requiring computation of an external memory address of [the] first memory location" as recited in Claim 2.

For these reasons, the proposed *Amerson-Pickett* combination fails to disclose, teach, or suggest the Appellant's invention as recited in Claim 2 (and its dependent claims). For similar reasons, the proposed *Amerson-Pickett* combination fails to disclose, teach, or suggest the Appellant's invention as recited in Claim 12 (and its dependent claims).

Accordingly, the Appellant respectfully requests that the § 103 rejection of Claims 2-7 and 12-17 be withdrawn and that Claims 2-7 and 12-17 be passed to allowance.

Ground of Rejection 4: Claims 2-7 and 12-17 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over *Amerson* in view of U.S. Patent No. 5,706,224 to Srinivasan et al. ("*Srinivasan*")

Claims 2-7 and 12-17

Claim 2 recites:

> [a] pipelined microprocessor capable of detecting an instruction that loads data from a first memory location that was previously stored to, wherein the instruction is detected without requiring computation of an external memory address of said first memory location for the instruction.

The Examiner fails to establish that the proposed *Amerson-Srinivasan* combination discloses, teaches, or suggests all elements of Claim 2. In particular, the Examiner fails to establish that the proposed *Amerson-Srinivasan* combination discloses, teaches, or suggests a microprocessor capable of detecting an instruction "without requiring computation of an external memory address of [a] first memory location for the instruction."

Srinivasan recites a semiconductor device that includes both random access memory (RAM) and content addressable memory (CAM) portions. (*Abstract*). A search word is used and compared to data words in the CAM portion. (*Col. 1, Lines 53-57*). When the value of a data word matches the value of the search word, information associated with the data word may be stored in or retrieved from a CAM cell without computing the address of the information. (*Col. 1, Lines 59-67*).

In every embodiment of *Amerson*, the processor compares the actual memory addresses being accessed by load and store instructions. Therefore, computation of the external memory address is required in *Amerson*.

Similarly, *Srinivasan* fails to disclose, teach, or suggest detecting "an instruction that loads data from a first memory location that was previously stored to ... without requiring computation of an external memory address of [the] first memory location." First, the technique described in *Srinivasan* relates to storing and retrieving information to and from a CAM memory. Nothing in the cited portion of *Srinivasan* relates to detecting an instruction. More specifically, nothing in the cited portion of *Srinivasan* relates to detecting an "instruction that loads data from a first memory location that was previously stored to" without "requiring computation of an external memory address of said first memory location for the instruction" as recited in Claim 2.

Second, the technique described in *Srinivasan* relates specifically to content-addressable memory. The Examiner fails to cite any portion of *Amerson* indicating that *Amerson* uses content-addressable memory or that *Amerson* could be modified to use content-addressable memory. The Examiner also fails to cite any portion of *Srinivasan* indicating that the technique described in *Srinivasan* could be used with non-content-addressable memory.

For these reasons, the proposed *Amerson-Srinivasan* combination fails to disclose, teach, or suggest the Appellant's invention as recited in Claim 2 (and its dependent claims). For similar reasons, the proposed *Amerson-Srinivasan* combination fails to disclose, teach, or suggest the Appellant's invention as recited in Claim 12 (and its dependent claims).

Accordingly, the Appellant respectfully requests that the § 103 rejection of Claims 2-7 and 12-17 be withdrawn and that Claims 2-7 and 12-17 be passed to allowance.

Ground of Rejection 5: Claims 20 and 21 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 5,850,138 to Engebretsen et al. ("*Engebretsen*") in view of U.S. Patent No. 5,615,357 to Ball ("*Ball*")

Claims 20 and 21

Claim 20 recites a method for operating a pipelined microprocessor, which includes:

> detecting a first instruction that stores data to a first memory location, said first instruction comprising syntax for computing an effective address for said first memory location;
>
> detecting a second instruction that loads data from a second memory location, said second instruction comprising syntax for computing an effective address for said second memory location;
>
> determining said syntax for said first instruction and said syntax for said second instruction;
>
> using said syntax for said first instruction and said syntax for said second instruction to determine a relationship between said first memory location and said second memory location, without requiring computation of said effective address for said first memory location and without requiring computation of said effective address for said second memory location; and
>
> using said relationship to determine whether to perform one of said first instruction and said second instruction.

The Examiner fails to establish that the proposed *Engebretsen-Ball* combination discloses, teaches, or suggests all elements of Claim 20. In particular, the Examiner fails to establish that the proposed *Engebretsen-Ball* combination discloses, teaches, or suggests using a "syntax" for a first instruction and a "syntax" for a second instruction to "determine a relationship between [a] first memory location and [a] second memory location, without requiring computation of [an] effective address for said first memory location and without requiring computation of [an] effective address for said second memory location."

Engebretsen recites a processor with memory storage locations allocated at compile time that are used for storing variable length data items. (*Abstract*). An alias table containing the base addresses of the data items in memory is used to access the data items. (*Col. 2, Lines 59-66*).

Ball recites a technique for adapting execution-driven simulators so they can accept execution traces. (*Abstract*). A simulator may receive a trace file associated with an executed benchmark program. (*Col. 2, Lines 30-60*). The trace file may have "effective memory addresses" for memory access instructions in the benchmark program. (*Col. 2, Lines 43-46*). The effective memory addresses are used to simulate execution of the memory access instructions without requiring the simulator to compute the effective memory addresses. (*Col. 2, Line 61 – Col. 3, Line 6*).

The Examiner acknowledges that *Engebretsen* fails to disclose these elements of Claim 20. (*08/08/05 Office Action, Page 10, Section 27*). Instead, the Examiner relies on *Ball* as disclosing these elements of Claim 20.

Ball specifically recites calculating the "effective memory addresses" of "memory access instructions" for inclusion in the "trace file." The simulator then uses the effective memory addresses during simulation. It is impossible for the simulator of *Ball* to use the effective memory addresses unless some component of *Ball* first computes the effective memory addresses. It is also irrelevant which component of *Ball* actually calculates the effective memory addresses. The only issue is whether *Ball* operates "without requiring computation" of "effective addresses" for memory locations. *Ball* clearly states that the effective addresses for memory

locations are used by the simulator, meaning that some component of *Ball* had to compute those effective addresses.

Moreover, *Ball* does not use the "syntax" associated with two instructions to determine a relationship between two memory locations without requiring computation of the effective addresses for the memory locations. *Ball* recites that the effective memory addresses are either taken from a trace file or computed and used. It is entirely unclear how a reference that computes and uses effective memory addresses can anticipate a claim that recites using the "syntax" associated with two instructions to determine a relationship between two memory locations without requiring computation of the effective addresses for the memory locations.

Because of this, both *Engebretsen* and *Ball* fail to disclose, teach, or suggest using a "syntax" for a first instruction and a "syntax" for a second instruction to "determine a relationship between [a] first memory location and [a] second memory location, without requiring computation of [an] effective address for said first memory location and without requiring computation of [an] effective address for said second memory location" as recited in Claim 20.

For these reasons, the proposed *Engebretsen-Ball* combination fails to disclose, teach, or suggest the Appellant's invention as recited in Claim 20 (and its dependent claims).

Accordingly, the Appellant respectfully requests that the § 103 rejection of Claims 20 and 21 be withdrawn and that Claims 20 and 21 be passed to allowance.

REQUESTED RELIEF

The Board is respectfully requested to reverse the outstanding rejections and return this application to the Examiner for allowance.

The Commissioner is hereby authorized to charge any fees connected with this communication (including any extension of time fees) or credit any overpayment to Munck Butrus Deposit Account No. 50-0208.

Respectfully submitted,

MUNCK BUTRUS, P.C.

Date: 6-14-2007

Daniel E. Venglarik
Registration No. 39,409
Attorney for Appellant

P.O. Drawer 802432
Dallas, Texas 75380
Phone: (972) 628-3600
Fax: (972) 628-3616
E-mail: *wmunck@munckbutrus.com*

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	David L. Isaman
Serial No.:	09/443,160
Filed:	November 19, 1999
For:	SYMBOLIC STORE-LOAD BYPASS
Technology Center:	2100
Group No.:	2183
Examiner:	Daniel H. Pan

APPENDIX A - Claims Appendix

1. (Cancelled).

2. A pipelined microprocessor capable of detecting an instruction that loads data from a first memory location that was previously stored to, wherein the instruction is detected without requiring computation of an external memory address of said first memory location for the instruction.

3. A pipelined microprocessor as claimed in Claim 2 wherein said pipelined microprocessor is capable of detecting an instruction that stores data into a second memory location that was previously read from without computing an external memory address of said second memory location.

4. A pipelined microprocessor as claimed in Claim 2 wherein said pipelined microprocessor is capable of detecting instructions that load data from identical memory locations that were previously stored to without computing external memory addresses of said identical memory locations.

5. A pipelined microprocessor as claimed in Claim 2 wherein said pipelined microprocessor is capable of detecting instructions that store data into identical memory locations that were previously read from without computing external memory addresses of said identical memory locations.

6. A pipelined microprocessor as claimed in Claim 4 wherein said pipelined microprocessor is capable of examining symbolic structure of said instructions that load data from identical memory locations that were previously stored to, and capable of detecting said instructions that load data from identical memory locations by examining said symbolic structure.

7. A pipelined microprocessor as claimed in Claim 5 wherein said pipelined microprocessor is capable of examining symbolic structure of said instructions that store data into identical memory locations that were previously read from, and capable of detecting said instructions that store data into identical memory locations by examining said symbolic structure.

8. A pipelined microprocessor as claimed in Claim 6 wherein said pipelined microprocessor is capable of detecting said instructions that load data from identical memory locations by identifying an identical offset address value from an identical base address value in a register within said pipelined microprocessor.

9. A pipelined microprocessor as claimed in Claim 7 wherein said pipelined microprocessor is capable of detecting said instructions that store data into identical memory locations by identifying an identical offset address value from an identical base address value in a register within said pipelined microprocessor.

10. A pipelined microprocessor as claimed in Claim 6 wherein said pipelined microprocessor comprises:

an instruction decode stage capable of detecting said instructions that load data from identical memory locations by identifying an identical offset address value from an identical base address value in a register within said pipelined microprocessor; and

a bypass element capable of sending a bypass signal to an instruction execution stage of said pipelined microprocessor that indicates that said instructions refer to an identical memory location.

11. A pipelined microprocessor as claimed in Claim 7 wherein said pipelined microprocessor comprises:

an instruction decode stage capable of detecting said instructions that store data into identical memory locations by identifying an identical offset address value from an identical base address value in a register within said pipelined microprocessor; and

a bypass element capable of sending a bypass signal to an instruction execution stage of said pipelined microprocessor that indicates that said instructions refer to an identical memory location.

12. A method for operating a pipelined microprocessor, said method comprising:

detecting in said pipelined microprocessor an instruction that loads data from a first memory location that was previously stored to, wherein the instruction is detected without requiring computation of an external memory address of said first memory location for the instruction.

13. A method for operating a pipelined microprocessor as claimed in Claim 12, said method further comprising:

detecting in said pipelined microprocessor an instruction that stores data into a second memory location that was previously read from without computing an external memory address of said second memory location.

14. A method for operating a pipelined microprocessor as claimed in Claim 12, said method further comprising:

detecting in said pipelined microprocessor instructions that load data from identical memory locations that were previously stored to without computing external memory addresses of said identical memory locations.

15. A method for operating a pipelined microprocessor as claimed in Claim 12, said method further comprising:

detecting in said pipelined microprocessor instructions that store data into identical memory locations that were previously read from without computing external memory addresses of said identical memory locations.

16. A method for operating a pipelined microprocessor as claimed in Claim 14, said method further comprising:

examining in said pipelined microprocessor symbolic structure of said instructions that load data from identical memory locations that were previously stored to; and

detecting said instructions that load data from identical memory locations by examining said symbolic structure.

17. A method for operating a pipelined microprocessor as claimed in Claim 15, said method further comprising:

examining in said pipelined microprocessor symbolic structure of said instructions that store data into identical memory locations that were previously read from; and

detecting said instructions that store data into identical memory locations by examining said symbolic structure.

18. A method for operating a pipelined microprocessor as claimed in Claim 16, said method further comprising:

detecting in an instruction decode stage of said pipelined microprocessor said instructions that load data from identical memory locations by identifying an identical offset address value from an identical base address value in a register within said pipelined microprocessor; and

sending a bypass signal from a bypass element to an instruction execution stage of said pipelined microprocessor wherein said bypass signal indicates that said instructions refer to an identical memory location.

19. A method for operating a pipelined microprocessor as claimed in Claim 17, said method further comprising:

detecting in an instruction decode stage of said pipelined microprocessor said instructions that store data into identical memory locations by identifying an identical offset address value from an identical base address value in a register within said pipelined microprocessor; and

sending a bypass signal from a bypass element to an instruction execution stage of said pipelined microprocessor wherein said bypass signal indicates that said instructions refer to an identical memory location.

20. A method for operating a pipelined microprocessor, said method comprising:

detecting a first instruction that stores data to a first memory location, said first instruction comprising syntax for computing an effective address for said first memory location;

detecting a second instruction that loads data from a second memory location, said second instruction comprising syntax for computing an effective address for said second memory location;

determining said syntax for said first instruction and said syntax for said second instruction;

using said syntax for said first instruction and said syntax for said second instruction to determine a relationship between said first memory location and said second memory location, without requiring computation of said effective address for said first memory location and without requiring computation of said effective address for said second memory location; and

using said relationship to determine whether to perform one of said first instruction and said second instruction.

21. A method for operating a pipelined microprocessor as claimed in Claim 20 wherein said syntax for said first instruction and said syntax for said second instruction refer to an identical memory location.

DOCKET NO. 98-MET-069C1 **PATENT**
CLIENT NO. STMI01-01012
Customer No. 30425

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	David L. Isaman
Serial No.:	09/443,160
Filed:	November 19, 1999
For:	SYMBOLIC STORE-LOAD BYPASS
Technology Center:	2100
Group No.:	2183
Examiner:	Daniel H. Pan

APPENDIX B
Evidence Appendix

none

DOCKET NO. 98-MET-069C1 **PATENT**
CLIENT NO. STMI01-01012
Customer No. 30425

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	David L. Isaman
Serial No.:	09/443,160
Filed:	November 19, 1999
For:	SYMBOLIC STORE-LOAD BYPASS
Technology Center:	2100
Group No.:	2183
Examiner:	Daniel H. Pan

APPENDIX C
Related Proceedings Appendix

none